UCRL-TR-204392

LAWRENCE
LIVERMORE
NATIONAL
LABORATORY

LSST data pipeline prototyping plans and strategy

Ghaleb M. Abdulla, Jim Brase, Kem Cook, Marcus Miller

5/9/2004

LSST Data Management Kickoff Meeting
May 10-11
Tuscon, AZ

LSST data pipeline prototyping plans and strategy

Ghaleb Abdulla, Jim Brase, Kem Cook, and Marcus Miller
Lawrence Livermore National Lab

Abstract

In this document we describe our approach and strategy for building the prototype for the image-stream analysis data pipeline. We start by describing the main research areas upon which we will be focusing; we then describe our plans on how to carry these research ideas to implement the data pipeline.

Introduction

LLNL has been involved in wide-field, time-domain astronomy for over a decade because of its involvement in the Macho Project. One of the key features of the Macho Project was that survey data was reduced in near real time and microlensing events were identified as they were happening. This allowed identification of non-standard events and their detailed study by networks of telescopes. LLNL continues its interest in astrophysical transients pursuing gamma-ray bursts, planetary transits, microlensing and more. This science driver is matched by an interest of the computer science community at LLNL in real time analysis systems and the research challenges they pose. This interest is driven not just by astrophysics, but a range of real time computing problems involving large sensor networks and the need for real time response and analysis. Because of this synergy, LLNL is interested in the real-time aspects of the proposed LSST pipeline. We hope that our interest and proposed activities will fulfill a need for a large institutional partner to address the real time pipeline.

Prototype system development

A potential approach to the high-level structure of the LSST data management is shown in Figure 1. The camera system delivers raw image data to the image conditioning pipeline, which in turn feeds co-add, object detection, and transient detection pipelines. The specific pipeline structure shown here is just an example but this class of architecture will be required. These real-time pipelines are large-scale parallel computer systems that are probably located near the telescope to accommodate the large communication bandwidth requirements. The pipelines feed their results into a set of databases for conditioned images, and object and transient catalogs.

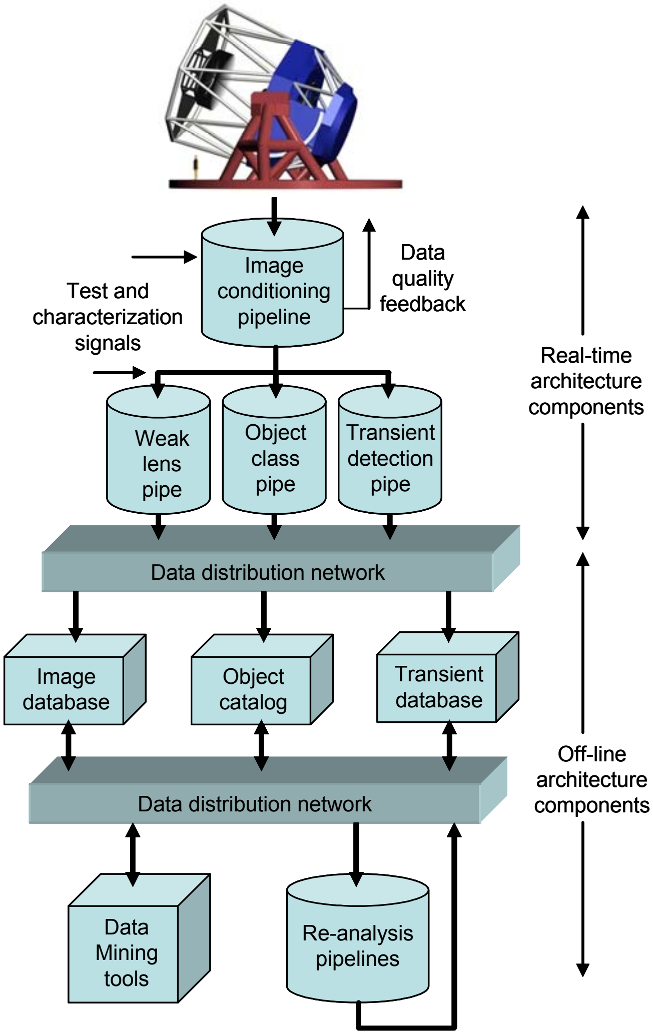


Image conditioning pipeline
Data quality feedback
Test and characterization signals
Weak lens pipe
Object class pipe
Transient detection pipe
Real-time architecture components
Data distribution network
Image database
Object catalog
Transient database
Data distribution network
Off-line architecture components
Data Mining tools
Re-analysis pipelines

Figure 1. Example of an overall LSST data management architecture. LLNL's real-time pipeline prototype will focus on the real-time architecture components.

The focus of data management work at LLNL will be on demonstrating the architectures and technology required for the real-time pipelines. These efforts will be coordinated with collaborators at NCSA and NOAO to develop a comprehensive LSST prototype data management environment.

Our objective is to develop a prototype hardware-software system that will define a viable path to the LSST real-time analysis pipelines and provide an opportunity to do pre-LSST science in the areas of transient event detection and moving object detection and tracking. We will develop a software infrastructure that allows flexible development of multiple communicating image analysis pipelines running in a cluster-based parallel hardware environment. Implementation of this testbed on LLNL's MCR cluster (Fig. 2) will allow us to scale data rates and quantities up to LSST levels to establish performance at those levels.

We will develop a modular approach to the software architecture that allows us to separate the analysis algorithms from the data objects in the system. This approach will provide for evolution of the algorithms and growth in capability throughout the system lifecycle. The system design will provide facilities for insertion of test data (signals and noise) for system validation and performance analysis. We will use available precursor data sets such as Macho[ζ] and SuperMacho[ζ] to demonstrate system performance and scaling and to develop and demonstrate the new approaches to time-domain astronomy that LSST will enable.

Research areas

In this section we will describe several of the computer science R&D areas that will be addressed using the LSST prototype pipeline

1. Modular Data-Centric Pipeline (MDCP)

In the LSST modular data centric pipeline the emphasis is on data objects and "intelligent data". Data objects will be stored accompanied with their metadata; consequently objects can be treated as application independent entities. The FITS data format supports the concept of metadata since it uses the header part of the image file to describe the contents. Algorithms will be decoupled from the data objects; new algorithms will be developed and implemented as functions. Legacy code will be wrapped with an interface to make them look like functions.

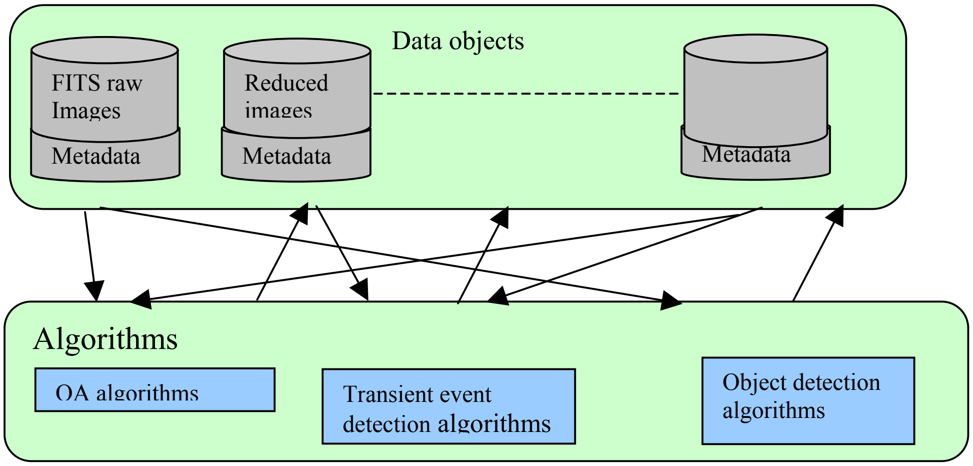


Figure 2. Decoupling the data objects from the algorithms will be a necessary step in building a data-centric pipeline

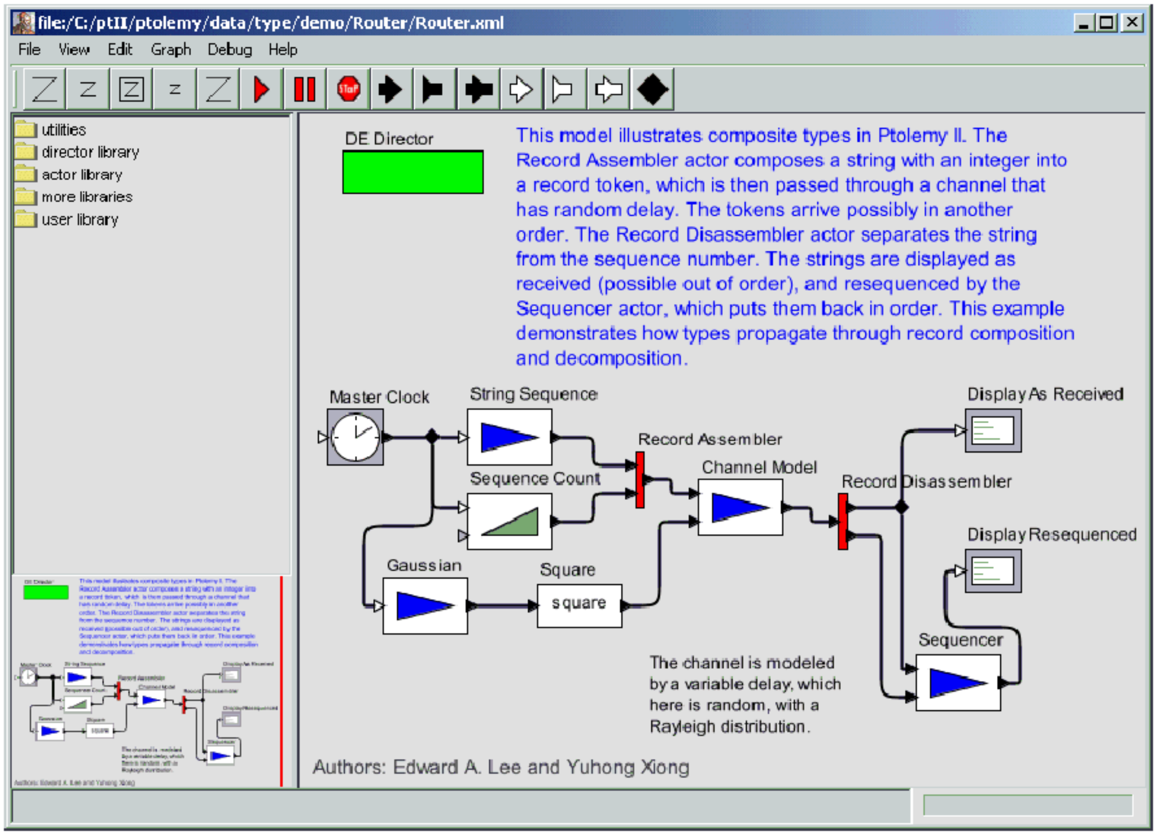
There are several benefits to this approach; some of these benefits are [1]:

- Stronger type checking: Current Super Macho data pipelines use a set of algorithms that are normally called using Perl or python scripts to process the data by reading files, processing the data within the file, and writing it into another file. Functions' signatures and metadata can be used for error checking and code sharing

[ζ] Macho and SuperMacho are used here as an example we can also use a data pipeline such as Loneos

- Support for interactive query processing: Interactive query processing is hard to support using the current data pipeline implementations. The code for the data pipeline has to be changed in order to support the interactive capabilities. However, using a data centric approach the function parameters can be changed while running the algorithm and the new results can be viewed immediately.
- Data provenance: In this model data provenance can be supported much easier since each function can write the necessary information about when and how the data was generated into a database or associate it with the data object written to a file as part of its metadata. The infrastructure to write such information will be part of the data pipeline design. Hence, data provenance can be easily supported in this environment.
- Quick prototyping of new data pipelines will be done quicker. Using the libraries of algorithms and an infrastructure that supports the algorithms integration and communication, new data pipelines that serve a new analysis model can be created easily. The infrastructure can support other application domains in addition to astrophysics.

2. Scientific Workflows



This model illustrates composite types in Ptolemy II. The Record Assembler actor composes a string with an integer into a record token, which is then passed through a channel that has random delay. The tokens arrive possibly in another order. The Record Disassembler actor separates the string from the sequence number. The strings are displayed as received (possible out of order), and resequenced by the Sequencer actor, which puts them back in order. This example demonstrates how types propagate through record composition and decomposition.

The channel is modeled by a variable delay, which here is random, with a Rayleigh distribution.

Authors: Edward A. Lee and Yuhong Xiong

Figure 3 Screen shot of Ptolemy that illustrates an example model

Using scientific workflows we can compose data pipelines using modular algorithms to perform the desired functionality. Figure 3 is an example of a system (Ptolemy) that uses the workflow concept to model a process. In this environment the algorithms are collected and classified as libraries, which can be navigated using the standard file hierarchy (shown at the left of the figure). The library can be easily extended by adding specialized algorithms as *actors*. These actors can be attached to each other in a way to achieve a final goal. On the right side of Figure 3 we see the result of composing several actors to decompose and compose a series of characters in the correct order. The library of algorithms can be extended to include a set of algorithms to process LSST data. A data pipeline to perform a specific functionality can be easily created using these algorithms. In addition if a scientist wants to see intermediate results then they can do that easily by using one of the plotting or output actors. The parameters to any of the algorithms can be changed while the data pipeline is running and the effects can be seen immediately.

3. Support for real-time query processing

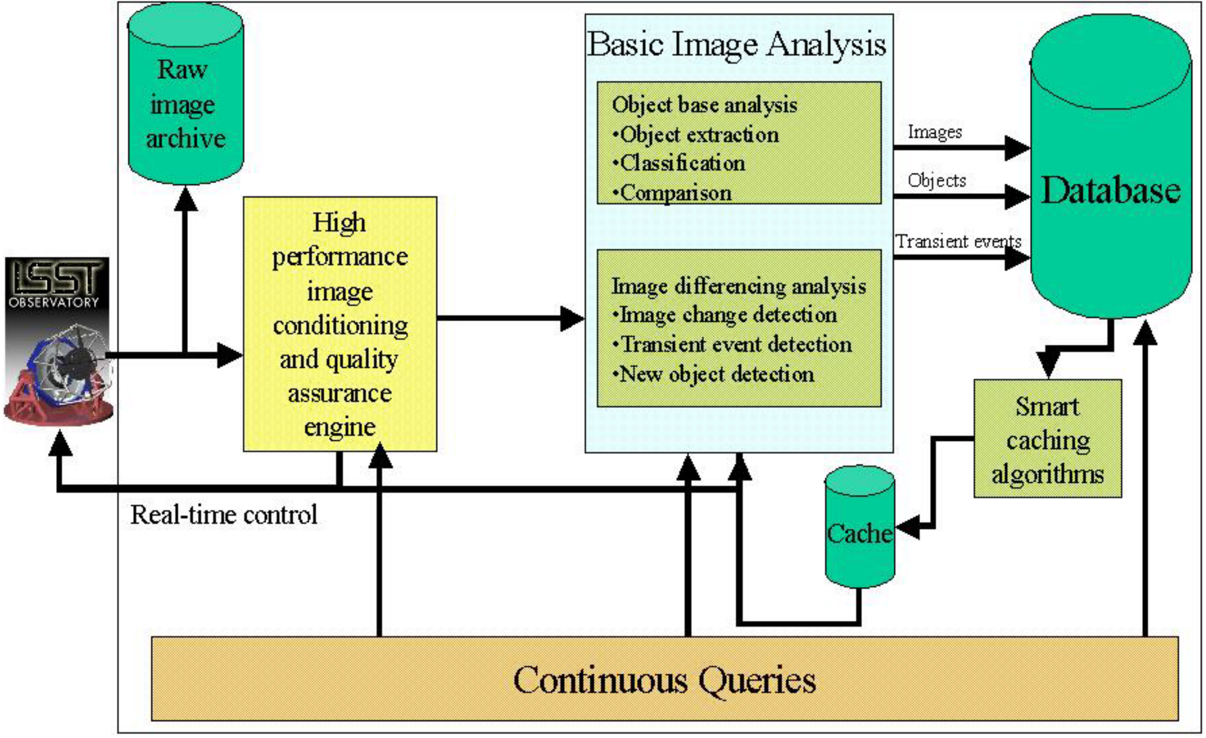


Figure 4 A real-time image stream data pipeline with support for continuous queries

Image analysis and data mining algorithms, whether local to the image or spatially or temporally global over several images, will be an important activity. Real time application of such algorithms is also of great importance. The real time data analysis can appear at the early stage in the pipeline (the high performance image conditioning and quality assurance engine in Figure 4) where the algorithms have to meet hard real-time requirements (on the order of seconds for huge data rates). Real time data analysis can also appear at a later stage in the prototype (the basic image analysis rectangle in Figure 4), where the real-time requirements can be less strict. Identifying moving or transient objects and new and interesting changes to previously characterized objects and as soon as they happen are two important requirements. To satisfy these requirements we will need to implement or integrate some of the identified data reduction and analysis algorithms. We will also investigate methods of using streaming data algorithms to support the basic data analysis requirements. Specifically, we will look into the query language that is needed to support *continuous* queries on image data and how can we register and support new queries. Supporting new queries will be an interesting and challenging problem. Since one of the major challenges is the performance of the system, new and registered queries have to execute in an optimized mode.

4. H/W and S/W cluster architecture

Mapping current astrophysics data pipelines to cluster architecture is a straightforward task. However, we can do better if we have a more complete understanding of the algorithms and we can exploit global parallel file systems properly. In addition, support for real time queries will be an interesting challenge in a distributed architecture. To answer a query, a considerable degree of communication must take place between the different nodes that are responsible for retrieving the answer. Query optimization will be a focus of our research since we want to support query processing in a real time distributed environment.

The overall H/W and S/W architecture design will be strongly impacted by the telescope site requirements. If LSST is deployed, as seems likely, on a remote mountain location, then the high bandwidth, and low latency communication requirements for both the archival digital image library and online database must be considered. We will explore architectures that will support the partitioning of processing effort between the on mountain software and hardware, and those functions which must be made available to the general scientific community, including image retrieval and queries based on specific content. Both cluster and grid based computing will play a vital role in delivering the overall performance targets for the LSST data pipeline.

Development strategy and plan

Project final goal:

Develop a data-centric modular image-streaming pipeline capable of handling data according to LSST real-time requirements and use available precursor data sets such as Macho and SuperMacho to demonstrate system performance and scaling and to develop and demonstrate the new approaches to time-domain astronomy that LSST will enable.

Short time goals:

1. Develop an initial version of the prototype that runs on a cluster machine using current data sets. To achieve this goal we will be performing the following activities:
   - Study and understand the current available data pipelines.
   - Port current data pipelines into one of the cluster machines (for example, MCR).
   - Run and perform timing experiments for the different algorithms.
   - Develop a simple parallel implementation of the current algorithms using MPI and using the current implemented parallelization model

2. Create a design document for the targeted LSST data pipeline. The design document will be created through brain storming sessions where scientists and prototype developers will identify:
   - LSST data and real-time requirements.
   - Goals that the system should meet. These goals are related to the users, the system (LSST prototype), and the system maintenance.
   - Assumptions and dependencies that might have an effect on the final system. We will be working under a number of assumptions due to the futuristic nature of the project. We will identify and list these assumptions in order to respond in the event of a technology change or breakthrough.
   - General system constraints and limitations and their impact on the system. These constraints will be the main source for the challenging problems that we need to address.

The design document will guide the development of the real-time data pipeline prototype. The prototype design and development activities will be guided by the LSST Data Management Scientist and Engineer. They will also be done with full communication and coordination with other LSST data management activities. We will be constantly interacting with the precursor projects, which will likely be where most algorithm development will occur and incorporate these developments into our prototype system. We will also closely interact with the database development activities so that the pipeline-database interface is well designed and understood.

References

Liu, David T. and Michael J. Franklin “GridDB: A Data-Centric Overlay for Scientific Grids” Tech Report, UCB//CSD-04-1311, 3/16/04